**Практична робота №1**

**Тема:** Робота на парі

**Мета:** Навчитись застосовувати розгалуження if…elif…else. Цикл while. Цикл for.

**Хід роботи**

**Завдання 1.**

Напишіть програму на Python, щоб надрукувати наступний рядок у певному форматі (див. вихідні дані).

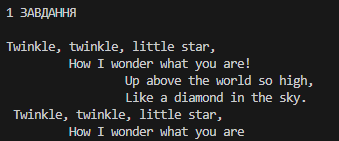
Зразок рядка : "Twinkle, twinkle, little star, How I wonder what you are! Up above the world so high, Like a diamond in the sky. Twinkle, twinkle, little star, How I wonder what you are"

**Код:**

print("\n1 ЗАВДАННЯ\n")

print("Twinkle, twinkle, little star, \n \t How I wonder what you are! \n \t \t Up above the world so high, \n \t \t Like a diamond in the sky. \n Twinkle, twinkle, little star, \n \t How I wonder what you are")

**Результат:**



**Пояснення:**

* print — функція, яка виводить текст на екран.
* \n — спеціальний символ для переносу на новий рядок.
* \t — спеціальний символ для табуляції (додає відступ).

**Завдання 2.**

Напишіть програму на Python, яка приймає радіус кола від користувача та обчислює площу

**Код:**

import math

print("\n2 ЗАВДАННЯ\n")

def find\_area(radius):

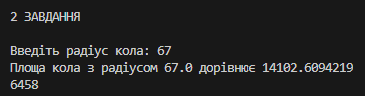
    return math.pi \* radius \* radius

radius = float(input("Введіть радіус кола: "))

area = find\_area(radius)

print(f"Площа кола з радіусом {radius} дорівнює {area}")

**Результат:**



**Пояснення:**

* import math — імпортує бібліотеку math, яка містить математичні функції та константи.
* print("\n2 ЗАВДАННЯ\n") — виводить заголовок завдання з новими рядками для форматування.
* Функція find\_area(radius):
  + Приймає аргумент radius.
  + Обчислює площу кола за формулою 𝜋×𝑟𝑎𝑑𝑖𝑢𝑠2*π*×*radius*2, використовуючи math.pi для значення 𝜋*π*.
* radius = float(input("Введіть радіус кола: ")):
  + Запитує у користувача введення радіуса.
  + Перетворює введене значення в число з плаваючою комою (тип float).
* area = find\_area(radius) — викликає функцію find\_area для обчислення площі кола з введеним радіусом.
* print(f"Площа кола з радіусом {radius} дорівнює {area}") — виводить результат у зручному для читання форматі.

**Завдання 3.**

Напишіть програму на Python для відображення першого та останнього кольорів із наступного списку

color\_list = ["Red","Green","White" ,"Black"]

**Код:**

print("\n3 ЗАВДАННЯ\n")

color\_list = ["Red", "Green", "White", "Black"]

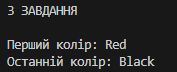
first\_color = color\_list[0]

last\_color = color\_list[-1]

print(f"Перший колір: {first\_color}")

print(f"Останній колір: {last\_color}")

**Результат:**



**Пояснення:**

* color\_list = ["Red", "Green", "White", "Black"] — створює список кольорів.
* first\_color = color\_list[0] — присвоює змінній first\_color перший елемент списку (індекс 0).
* last\_color = color\_list[-1] — присвоює змінній last\_color останній елемент списку (індекс -1).
* print(f"Перший колір: {first\_color}") — виводить перший колір зі списку.
* print(f"Останній колір: {last\_color}") — виводить останній колір зі списку.

**Завдання 4.**

Напишіть програму на Python, яка приймає ціле число (n) і обчислює значення n+nn+nnn

Вибіркове значення n дорівнює 5

Очікуваний результат : 615

**Код:**

print("\n4 ЗАВДАННЯ\n")

n = int(input("Введіть ціле число: "))

n\_str = str(n)

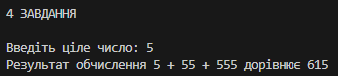
nn = int(n\_str \* 2)

nnn = int(n\_str \* 3)

result = n + nn + nnn

print(f"Результат обчислення {n} + {nn} + {nnn} дорівнює {result}")

**Результат:**



**Пояснення:**

* Запитує в користувача ціле число.
* Зберігає введене число у змінній n.
* Перетворює число n в рядок (str).
* Потім об'єднує два рядки n\_str тричі (двічі та тричі), конвертує ці рядки у цілі числа (int), і зберігає їх у змінних nn та nnn відповідно.
* Після чого обчислює суму чисел n, nn та nnn і зберігає результат у змінну result.
* Нарешті, виводить результат у вигляді рядка, використовуючи форматованний рядок (f-string), де вказує введене число n, подвійне та потрійне його повторення, а також результат їх суми.

**Завдання 5.**

Напишіть програму на Python, щоб друкувати всі парні числа із заданого списку чисел в однаковому порядку та зупиняти друк, якщо в послідовності є числа, які йдуть після 237

numbers = [386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978, 328, 615, 953, 345, 399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950, 626, 949, 687, 217, 815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81, 379, 843, 831, 445, 742, 717, 958,743, 527]

**Код:**

print("\n5 ЗАВДАННЯ\n")

numbers = [

386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978, 328, 615,

953, 345, 399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950,

626, 949, 687, 217, 815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81,

379, 843, 831, 445, 742, 717, 958, 743, 527]

for number in numbers:

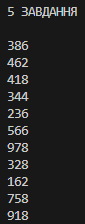
if number == 237:

break

if number % 2 == 0:

print(number)

**Результат:**



**Пояснення:**

* Створює список numbers, який містить різні цілі числа.
* Перебирає кожний елемент списку numbers.
* Перевіряє, чи поточний елемент дорівнює числу 237. Якщо так, то виходить з циклу for за допомогою команди break.
* Якщо поточний елемент парне число (перевірка за допомогою % 2 == 0), то виводить його.

**Завдання 6.**

Напишіть програму на Python, щоб перевірити, чи за кожною послідовною послідовністю нулів слідує послідовна послідовність одиниць однакової довжини в даному рядку. Повертає true/false

Послідовність: 01010101

Результат: True

Послідовність: 00011100011

Результат: False

**Код:**

print("\n6 ЗАВДАННЯ\n")

number\_list = ["01010101", "00011100011"]

for number in number\_list:

valid = True

i = 0

n = len(number)

while i < n:

if number[i] == '0':

zero\_count = 0

while i < n and number[i] == '0':

zero\_count += 1

i += 1

one\_count = 0

while i < n and number[i] == '1':

one\_count += 1

i += 1

if zero\_count != one\_count:

valid = False

break

else:

i += 1

print(valid)

**Результат:**



**Пояснення:**

* Має список рядків number\_list, що містить рядки, які представляють собою послідовності нулів та одиниць.
* Перебирає кожний рядок number в number\_list.
* Ініціалізує змінну valid на початку кожної ітерації циклу як True, що означає, що рядок є валідним.
* Встановлює лічильник i на 0 та визначає довжину поточного рядка number.
* Проводимо перевірку для кожного символу у рядку number.
* Якщо поточний символ - '0', то знаходить кількість нулів підряд (zero\_count), а потім кількість одиниць підряд (one\_count).
* Порівнює кількість нулів та одиниць. Якщо вони не рівні, то рядок не валідний (valid стає False) та виходить із внутрішнього циклу.
* Якщо поточний символ - '1', просто збільшує i та переходить до наступного символу.
* На кожній ітерації виводить значення valid, яке вказує, чи є рядок валідним.

**Завдання 7.**

Вивести усі парні числа від -n до n при тому що перебір буде здійснюватися із кроком m. де m - порядковий номер студента у журналі

**Код:**

print("\n7 ЗАВДАННЯ\n")

n = int(input("Введіть значення n: "))

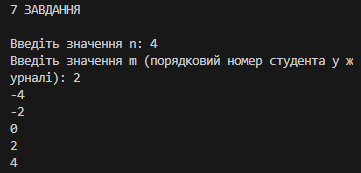
m = int(input("Введіть значення m (порядковий номер студента у журналі): "))

for i in range(-n, n+1, m):

if i % 2 == 0:

print(i)

**Результат:**



**Пояснення:**

* Запитує в користувача значення n (яке є цілим числом).
* Запитує в користувача значення m, яке є порядковим номером студента у журналі.
* Проводить ітерацію в межах від -n до n з кроком m.
* Перевіряє, чи поточне значення i є парним числом за допомогою оператора % 2 == 0.
* Якщо поточне значення i є парним, то виводить його.

**Завдання 8.**

Порахувати Кількість можливих комбінацій при введені паролю кожна комірка якого може містити один із 25 варіантів символів при цьому всі інші

комірки не можуть містити того ж символа що уже використаний. кількість комірок m. де m - порядковий номер студента у журналі

**Код:**

print("\n8 ЗАВДАННЯ\n")

m = int(input("Введіть значення m (порядковий номер студента у журналі): "))

if m > 25:

print("m не може бути більше 25")

else:

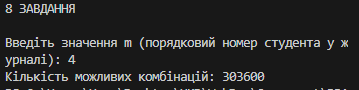
combinations = 1

for i in range(25, 25 - m, -1):

combinations \*= i

print(f"Кількість можливих комбінацій: {combinations}")

**Результат:**



**Пояснення:**

* Запитує в користувача значення m, яке є порядковим номером студента у журналі.
* Перевіряє, чи значення m більше за 25. Якщо так, виводить повідомлення, що m не може бути більше 25.
* Якщо значення m не перевищує 25, то встановлює змінну combinations рівною 1.
* Проводить ітерацію від 25 до (25 - m) включно з кроком -1.
* На кожній ітерації множить поточне значення combinations на поточне значення i.
* Після завершення циклу виводить кількість можливих комбінацій за допомогою рядка форматування f-string.

**Завдання 9.**

Створіть будь-який список.

а.) Розширте список, додавши до нього усі елементи списку L[3, 6, 7].

б.) Вставте на другий елемент значення 33333.

в.) Розташуйте список у зворотньому порядку.

г.) У кінець списку додайте 3.

д.) Видаліть перший елемент списку який має значення 3.

е.) Розташуйте список у порядку збільшення.

ж.) Очистить список.

**Код:**

print("\n9 ЗАВДАННЯ\n")

my\_list = [1, 2, 4, 5]

L = [3, 6, 7]

my\_list.extend(L)

my\_list.insert(1, 33333)

my\_list.reverse()

my\_list.append(3)

my\_list.remove(3)

my\_list.sort()

my\_list.clear()

print(my\_list)

**Результат:**



**Пояснення:**

* Ініціалізує список my\_list з початковими значеннями [1, 2, 4, 5].
* Розширює список my\_list, додаючи до нього всі елементи зі списку L, отримуючи [1, 2, 4, 5, 3, 6, 7].
* Вставляє число 33333 на позицію з індексом 1, тобто між елементами 1 та 2, отримуючи [1, 33333, 2, 4, 5, 3, 6, 7].
* Перевертає список, змінюючи порядок елементів, отримуючи [7, 6, 3, 5, 4, 2, 33333, 1].
* Додає число 3 в кінець списку, отримуючи [7, 6, 3, 5, 4, 2, 33333, 1, 3].
* Видаляє перше знайдене число 3 із списку, отримуючи [7, 6, 5, 4, 2, 33333, 1, 3].
* Сортує список в порядку зростання, отримуючи [1, 2, 3, 4, 5, 6, 7, 33333].
* Очищує список, змінюючи його на порожній список [].
* Виводить порожній список.

**Завдання 10.**

Напишіть програму для обліку читача у бібліотеці.

Облік здійснюється з допомогою словника. При вході в програму одразу виводиться інформація про користувача.

У словнику мають бути ключі:

- ID (номер по списку)

- прізвище;

- Ім’я

- Група

- Курс

- Книги (борг) - список

- Статистика книг – список

У списку борг зберігаються назви книг які взяв користувач у користування.

У списку статистика зберігаються усі книги які брав користувач і повернув (можуть повторюватися)

Функціонал:

Зробити цикл який буде працювати доки користувач не введе 0 (вихід з циклу)

Якщо користувач введе 1 то програма видає карту читача

Якщо користувач введе 2 то програма пропонує взяти книгу. Користувач має ввести назву книги після чого книга записується в борг.

Якщо користувач вводить 3 то програма пропонує повернути книгу (попередньо виводиться список книг у боргу). Якщо книг немає - виводиться відповідне повідомлення. Якщо книга відсутня у списку - виводиться відповідне повідомлення. Якщо книга є у списку то така книга видаляється зі списку борг і записується у список статистика.

Дані користувача це Ваші особисті дані (і мають збігатися з журналом)

**Код:**

print("\n10 ЗАВДАННЯ\n")

user = {

    "ID": 1,

    "Прізвище": "Бойчук",

    "Ім’я": "Богдан",

    "Група": "ІПЗс-21-1",

    "Курс": 3,

    "Книги (борг)": [],

    "Статистика книг": []

}

def print\_reader\_card(user):

    print("Карта читача:")

    print(f"ID: {user['ID']}")

    print(f"Прізвище: {user['Прізвище']}")

    print(f"Ім’я: {user['Ім’я']}")

    print(f"Група: {user['Група']}")

    print(f"Курс: {user['Курс']}")

    print(f"Книги (борг): {', '.join(user['Книги (борг)']) if user['Книги (борг)'] else 'Немає'}")

    print(f"Статистика книг: {', '.join(user['Статистика книг']) if user['Статистика книг'] else 'Немає'}")

print\_reader\_card(user)

while True:

    print("\nМеню:")

    print("1 - Показати карту читача")

    print("2 - Взяти книгу")

    print("3 - Повернути книгу")

    print("0 - Вийти")

    choice = input("Введіть ваш вибір: ")

    if choice == "0":

        break

    elif choice == "1":

        print\_reader\_card(user)

    elif choice == "2":

        book\_name = input("Введіть назву книги, яку хочете взяти: ")

        user["Книги (борг)"].append(book\_name)

        print(f"Книга '{book\_name}' додана до боргу.")

    elif choice == "3":

        if not user["Книги (борг)"]:

            print("У вас немає книг у боргу.")

        else:

            print(f"Книги у боргу: {', '.join(user['Книги (борг)'])}")

            book\_name = input("Введіть назву книги, яку хочете повернути: ")

            if book\_name in user["Книги (борг)"]:

                user["Книги (борг)"].remove(book\_name)

                user["Статистика книг"].append(book\_name)

                print(f"Книга '{book\_name}' повернута.")

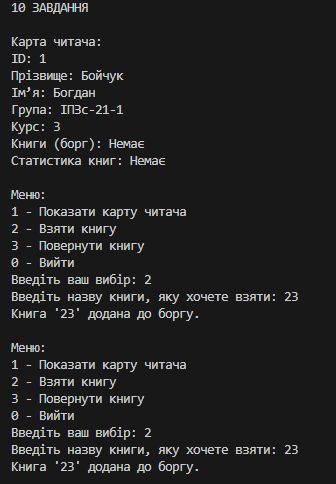
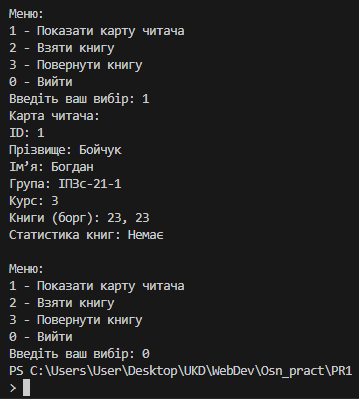
            else:

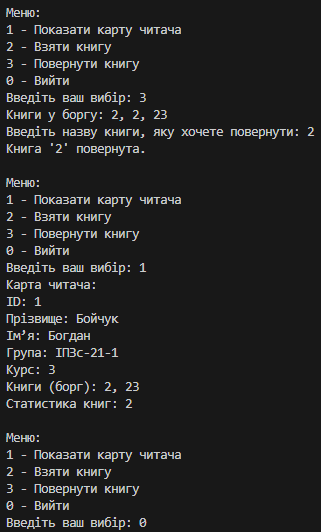
                print(f"Книга '{book\_name}' відсутня у вашому боргу.")

    else:

        print("Невірний вибір. Спробуйте ще раз.")

**Результат:**



**Пояснення:**

* Створюється словник **user**, що містить інформацію про читача.
* Визначається функція **print\_reader\_card**, яка виводить на екран інформацію про читача.
* Запускається цикл **while True**, який працює доти, доки користувач не обере вихід (**0**).
* У циклі користувачеві пропонується меню з опціями:
  + Показати карту читача
  + Взяти книгу
  + Повернути книгу
  + Вийти
* Залежно від обраної опції, виконуються певні дії:
  + Виводиться інформація про читача.
  + Додається книга до списку боргованих книг.
  + Повертається книга зі списку боргованих книг.
  + Вихід з програми.
* Код має валідацію для неправильного вводу вибору опцій та обробку ситуацій, коли користувач намагається повернути книгу, якої немає у його боргу.